The National Virtual Observatory Book
ASP Conference Series, Vol. 382, © 2008
M. J. Graham, M. J. Fitzpatrick, and T. A. McGlynn, eds.

Chapter 11: Web-based Tools—VO Region Inventory Service

John C. Good

Introduction

As the size and number of datasets available through the VO grows, it becomes increasingly critical to have services that aid in locating and characterizing data pertinent to a particular scientific problem. At the same time, this same increase makes that goal more and more difficult to achieve. With a small number of datasets, it is feasible to simply retrieve the data itself (as the NVO DataScope service does). At intermediate scales, "count" DBMS searches (searches of the actual datasets which return record counts rather than full data subsets) sent to each data provider will work. However, neither of these approaches scale as the number of datasets expands into the hundreds or thousands.

Dealing with the same problem internally, IRSA developed a compact and extremely fast scheme for determining source counts for positional catalogs (and in some cases image metadata) over arbitrarily large regions for multiple catalogs in a fraction of a second. To show applicability to the VO in general, this service has been extended with indices for all 4000+ catalogs in CDS Vizier (essentially all published catalogs and source tables).

In this chapter, we will briefly describe the architecture of this service, and then describe how this can be used in a distributed system to retrieve rapid inventories of all VO holdings in a way that places an insignificant load on any data supplier. Further, we show and this tool can be used in conjunction with VO Registries and catalog services to zero in on those datasets that are appropriate to the user's needs.

The initial implementation of this service consolidates custom binary index file structures (external to any DBMS and therefore portable) at a single site to minimize search times and implements the search interface as a simple CGI program. However, the architecture is amenable to distribution. The next phase of development will focus on metadata harvesting from data archives through a standard program interface and distribution of the search processing across multiple service providers for redundancy and parallelization.

1. Service Architecture

The key to fast access to information is proper indexing. For a typical relational database, this is provided by *B-Tree* indexing, which is essentially a fast way to find the closest match to an input value in a sorted list. B-Tree indices can be maintained easily and allow for record inserts and deletes, which make them perfect for a DBMS.

Unfortunately, B-Tree indices are one-dimensional in the sense that the variable being indexed must be linearly sortable. This makes them less than perfect for "geographical" data, where two- or three-dimensional position is the primary attributed on

which we wish to sort. In this case, the most common scheme is to use some sort of partitioning scheme where items in the database are allocated to a set of adaptable "bins". These can be fixed, hierarchical, regions on the sky (such as the *Hierarchical Triangular Mesh* scheme used by several VO sites to group catalog sources into nested triangular bins on the sky), or separable / combinable (and often overlapping) range boxes (such as the *R-Tree* index used by several Geographical Information System packages).

In any case, these multi-dimensional indices are used a little differently from the linear B-Trees in that they work by efficiently rejecting large subsets of the records and reducing the problem down to one of performing a secondary filtering on a small candidate list. In this step, the actual database records are retrieved and their positions (and other relational constraints) are checked for detailed matching.

For our purposes speed is paramount, so we have limited the search to just position and created an index file structure which allows us to utilize a 2D index search while still carrying along the exact locations of all sources (for the detailed positional filtering). In consequence, even a very large positional catalog (e.g. a billion records for 2MASS) can be searched for a count (and optionally a list of source IDs) for an arbitrary sky region in a few milliseconds. Since this approach is highly parallelizable, it easily scales to arbitrarily large datasets.

The index file structures as currently defined are not extensible; if the dataset changes, the index must be regenerated. In practice, this is not particularly onerous, though we may add extensibility in the future. All that is needed to generate an index is a complete list of coordinates (for a billion records this is only 16 GB) and the resultant index files are not much larger than this input data. The index files are independent of any DBMS and do not need to be associated with the originating DBMS or even organization. At the moment we are utilizing a commercial DBMS in part of the data processing, but plan to build a standalone and portable version of the toolkit.

2. Coverage Maps

Having the exact coordinates of all sources from a catalog in a system that allows fast positional searches, we can produce some interesting ancillary information. One issue that repeatedly arises concerns sky coverage: was a region observed as part of a survey and were any sources found. The first part requires survey "footprints", such as are being developed as part of the NVO Project at John's Hopkins University (see Chapters 9 and 10.

The second can be derived from the position database collected above. For every catalog we index, we also produce all-sky coverage maps; for example Aitoff projection maps at 15-arcmin resolution where the value of a pixel is the number of catalog sources contained within the pixel. When rendered in JPEG form, this provides a simple way for users to see catalog coverage on the sky.

The same maps can be used to determine which regions are in common between catalogs. For a set of catalogs, we can rapidly produce a composite map that shows the number of catalogs with data for a given pixel. This can take the form of a mask, e.g. only pixels with data from all (or only the selected) catalogs are turned on.

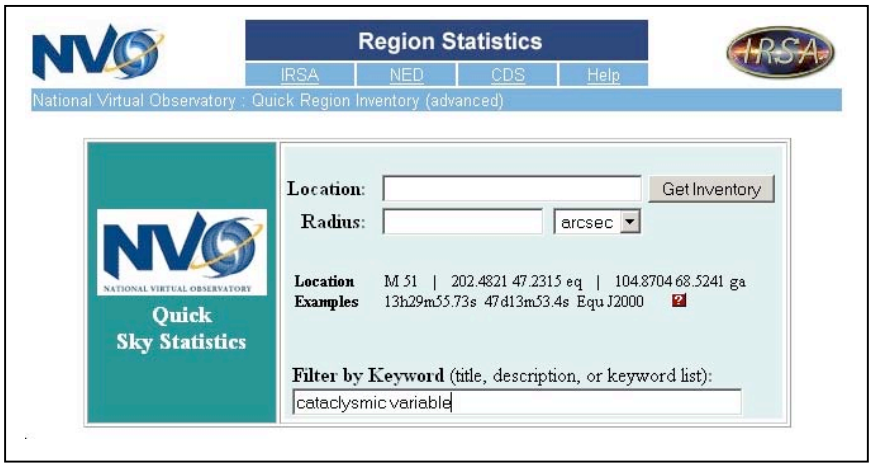

Figure 1. Search Form

3. Service

The NVO Region Statistics service (Figure 1) is an initial attempt to use this infrastructure to give the user an inventory of VO-accessible data. In this case the service is limited to catalog data available through IRSA, NED, and CDS. The user simply gives a location (coordinates or NED/SIMBAD object name) and a size (default 5 arcmin). The service then rapidly (3-4 seconds) determines all the catalogs with sources in this region and returns a table of results (Figure 2). Each record in this table contains one record for each catalog that has data, with the following fields:

- The count;
- A link to download the catalog subset itself;
- The catalog name/description (with a link to further documentation); and
- A thumbnail coverage map.

Besides the data download, there are two additional functions that can be accessed from this result page. Each list record also includes a checkbox, so a subset of catalogs can be selected. Then additional services can be invoked to either generate a composite coverage map (as described above) or to make a source map for the region of interest.

Obviously, coverage maps are most useful for identifying regions of the sky that have been covered by a uniform set of catalog surveys and source maps make the most sense for reasonably small regions.

4. Future Development

There are several ways in which this service can be augmented:

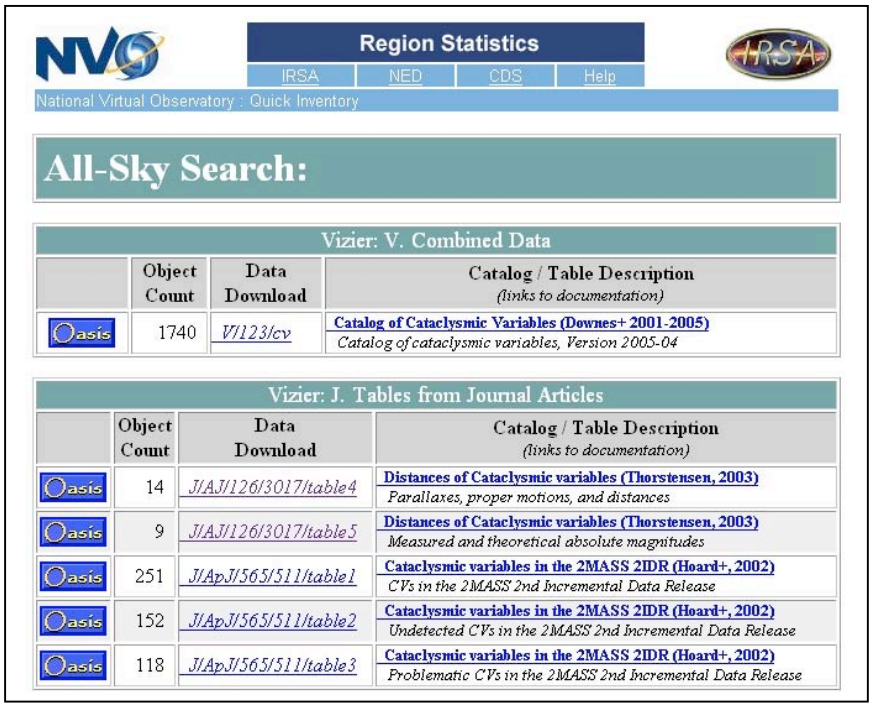

Figure 2. Search results.

- The current service focuses on catalog data but can be extended to include image and spectra metadata. For catalogs, one of the returned columns is a link to actually get the data. For images and spectra, that link would be to get the image/spectra metadata (e.g. from an SIA service) which then contains links to the individual images/spectra.

- The process of collecting positional information and making the count index structure is not automated. An update mechanism needs to be built which mines the VO Registry looking for new/updated datasets, downloads the positional information, and builds the index. Alternatively, data suppliers could streamline this by building the indices themselves and making them available. They could also mount an inventory service for their own data alone, but for operational efficiency, the indices should also be collected at a few central sites.

- The service should provide program-friendly access to the inventory information to allow general VO data mining. This can be in the form of a complete inventory in XML, a web service dialog (essentially a function library to get specific parts of the information), or a simple "script" (e.g. using `wget`) that can be run to download the data.

- The service should allow for an uploaded list of sources (by name and/or position in fairly free-form syntax). Since such lists can become quite long, this also necessitates a formal foreground/background notification mechanism.

- There is no one single way to present the inventory information. Especially when there is a long list of sources, the right approach is to collect the inventory information, then provide the user with various ways to view and query it (e.g. are there any catalogs which had matches for all of my input sources?).